project abstract:

CRUX PPC on pSeries

to attention of:

author: Nico M.
email: acrux@cruxppc.org
creation date: 30 March 2010

CRUX PPC - GNU/Linux on pSeries

IBM maintains strategic partnerships with both Red Hat and Novell for their enterprise Linux distributions. But what about Community distros?

Some of these seem to work well on pSeries servers and today is quite simple for everyone to get a second hand workstation or server equipped with one or more POWERn cpus.

Community distributions (as listed on IBM developerWorks: Linux for Power Architecture) are CRUX, Debian, Fedora, Gentoo and OpenSUSE and we are going to focus our attention to a comparatively marginal one: **CRUX for PowerPC**.

...a small digression about CRUX PPC non-profit project

CRUX PPC is a port of CRUX[1] for the *Power Architecture*® platform. It's a GNU[2] system with a Linux[3] kernel and runs on Apple NewWorld (both 32 and 64bit) PowerPC, YDL Powerstation, Genesi PegasosII and EFIKA, ACube Sam440ep, IBM RS/6000 CHRP and pSeries servers. **CRUX PPC** includes support for laptop-specific features, extended hardware support and server tools. Further info are available on *CRUX PPC Release Notes*.

CRUX PPC is *a volunteer driven non-profit project* based on the releases of CRUX for x86. Due to the small size of the team, project design and specification will be discussed with the whole team: glocalization is a CRUX PPC strong point.

It is targeted to the experienced Linux user. An user with average experience and fast internet connection will take some days to set up a general-purpose workstation system comfortable enough in daily usage. A server system might take less time but more attention. After the initial set up, **CRUX PPC** will be really easy to keep updated, and new software installation will be a pleasure either. If you're not familiar with a *NIX and GNU/Linux system, you will have the chance to learn a lot, but you will need some extra patience. To set up **CRUX PPC** properly you will have to get somehow familiar with PowerPC hardware too.

Unlike other distributions, the primary goal of CRUX is not to be the most popular (read: biggest number of users) or to put as many features as possible into the system. The idea is rather to build a streamlined distribution, targeted at a specific audience which in turn gets a system without compromises. The result may not be the best distribution for everybody, but as mentioned above, that's not the goal. If you're part of the target audience however, you might enjoy running CRUX just as much as we do.
It's quite hard to do an objective comparison of Linux distributions, since it's often a matter of taste. We therefore believe that you should compare potential candidates yourself, and choose the one you like best. One thing you should remember is that CRUX is targeted at experienced users who don't mind doing some work themselves, so if you're an completely new to Linux-based operating systems or want a zero-effort distribution, CRUX is not the optimal choice. Also, especially new users sometimes confuse a deliberately

omitted feature with a lack thereof and suggest it as an “improvement”; you do yourself and us a favour if you try to get yourself adjusted to CRUX for a while before making suggestions to seemingly improve the system; many of the developers have been working with various Linux distributions already and are quite aware of their features.

The **CRUX PPC** Project has been considered eligible to adopt and use Power Source Logo from Power.org. Power Source Logo identifies complementary products specifically designed for use with or usable with and services rendered in developing, building or maintaining cores, processors, and hardware and computer systems of any type that include and are 100% compatible with the core Power Architecture technology, specifically the standard instruction set architecture represented by the Power ISA or PowerPC ISA specifications. Products may include software, tools, and test equipment as examples. Services may include manufacturing and programming, as examples. May be used on collateral such as product or service brochures, fact sheets, user manuals and presentations, product packaging and on websites related to qualified products or services.

CRUX PPC is, as usual, released via two different installation ISO: 32bit and 64bit. The 32bit version is based on a single lib toolchain instead the 64bit one comes with a multilib toolchain and these two versions share the same ports tree. **CRUX PPC** ports repositories overlay is well managed with *ilenia*[4] a fast and powerful replacement to standard ports utility and a valid alternative to *prt-get* used in CRUX x86.

...we need to power-on our system

The installation on unmanaged systems is, perhaps, the most common scenario for a normal end user. Unmanaged is a server that doesn't have an Hardware Management Console (HMC) and has not been partitioned. Its tty output is displayed over a graphics card connected to a monitor or through a serial console.

The HMC is an Intel-based PC running IBM specialized software that when connected to a server allows you to partition and do various service-oriented activities as well as virtual serial sessions with the partitions.

If you are unable to access the Advanced System Management Interface (ASMI) using a network connection, you must attach an ASCII terminal to the service processor using a serial cable.

The ASCII terminal is connected to the server through a serial link. The ASCII interface to the ASMI provides a subset of the Web interface functions and it's available only when the system is in the platform standby state. Remember that it is not available during the Initial Program Load (IPL) or run time.

The console session will respond to communications running at 19200bps (9600bps for POWER3/POWER3-II/POWER4/POWER4+) with data bits of 8, a parity of None, and stop bits of 1. This connection to the serial connection port 1 (S1) using a 15-pin null-modem cable also allows you to access the System Management Services (SMS). Use the SMS menus to view information about your system and to perform steps such as changing the boot list and setting the network installation parameters.

When the system is in unmanaged mode and you power it on, the system goes through several levels of service checks and RTAS verifications. This is very time consuming. Once you have the server cabled properly and have powered it on, the Flexible Service Processor (FSP) will boot and output System Resource Codes (SRCs) to the control panel. The control panel is a small LCD on the front of the pSeries.

If this is the first time you have worked with the FSP, you will be asked to set a variety of passwords.

…and now we're ready for a fresh new installation on unmanaged system

If you have a good skill you'd be able to perform a **CRUX PPC** installation across your LAN or you can install it in a chroot environment from a different Linux distribution.

Anyway in order to install POWER-based systems from the network you need a DHCP server, a TFTP server and NFS or FTP or HTTP network installation source.

CRUX PPC installation (from CD-ROM) on pSeries is described on *CRUX PPC Handbook*. It didn't matter whether it was a standalone, LPAR, DLPAR or micropartitioned environment.

The CD should autoboot on your pSeries box, but sometimes it does not. In that case, you have to set up your cdrom as a bootable device in the multi-boot menu. If you start your machine with a monitor and a keyboard attached, you can reach the multi-boot menu pressing the F1 key on startup. But if you start your machine using the serial console, then you have to press 1.

Yaboot is loaded from this device and you are presented to the usual yaboot prompt. Select at the boot prompt the kernel *pseries* for RS64/POWERn cpu based machines (if you have a videocard) or *pseries-ttyS0,9600* if you are going to install through a serial console. **CRUX PPC** bootkernels have IBM GXT4000P/GXT6500P support built in too.

For newer systems (POWER5 or newer) you need to set the serial console at 19200bps thus, i.e., you must write *pseries root=/dev/hd?? console=ttyS0,19200* where *hd??* is the device of the CD-ROM Drive with the **CRUX PPC** CD-ROM inserted in (e.g. *hda* or *scd0*).

Login as root (no password required) and use fdisk or parted to create and format the partition(s) you want **CRUX PPC** to be installed on.

```
$ fdisk /dev/sd?
$ mkreiserfs /dev/sd??
$ mkswap /dev/sd??
```

Creating PC-Style partition table is recommended; cfdisk or fidks are suitable for this. You will need a special PReP Boot partition (type 41) of 8 megabytes.

The amount of disk space you need depends on how many packages you choose to install. I recommend having at least a 1GB root partition (**CRUX PPC** will use about 300MB-700MB depending on your configuration).

Using ReiserFS is recommended, but there is support for Ext2fs/Ext3fs/Ext4fs, XFS and

JFS as well. Further, I highly recommend separating system data from user data, i.e. use a separate partition for */home* (and possibly */var*) since that will make your life a lot easier the day you want to upgrade/reinstall/remove your system.

Mount the partition on which you want to install this distribution.

```
$ mount -t type /dev/sd?? /mnt
```

If you want the installation to span more than one partition, mount those partitions as well. For example, if you want to have a different partition for */home* or */var*, then do:

```
$ mkdir /mnt/var
$ mount -t type /dev/sd?? /mnt/var
```

Activate your swap partition(s).

```
$ swapon /dev/sd??
```

Type *setup* to start the package installation script. The script will ask where you mounted your new root partition and which packages you want to install. Just select the packages you want and nothing else will be installed. However, I recommend at least installing all the packages marked core.

Once it has installed the selected packages, the setup script will display an installation log. Make sure the last line in the log says "*0 error(s)*".

If you at a later stage find that you need some additional packages you can just mount the **CRUX PPC** CD-ROM and use pkgadd to install them.

Now it's time to compile your kernel and do basic system configuration. The kernel compilation requires that you "*chroot*" into your new **CRUX PPC** installation.

```
$ mount --bind /dev /mnt/dev
$ mount --bind /tmp /mnt/tmp
$ mount -t proc proc /mnt/proc
$ mount -t sysfs none /mnt/sys
$ chroot /mnt /bin/bash
```

You can type *setup-chroot* instead of issuing the previous commands to chroot in the system installed in */mnt* .

Set the root password.

```
$ passwd
```

Edit */etc/fstab* to configure your filesystem(s). Editors vim, nano and mcedit are available.

If you want access your computer via serial console you also have to add *ttyS0* in the */etc/securetty* and *c7:2:respawn:/sbin/agetty 9600 ttyS0* (*c7:2:respawn:/sbin/agetty 19200 ttyS0)* in the */etc/inittab* file. IBM System p and PPC970 cpu based systems work with 19200bps serial console.

Edit */etc/rc.conf* to configure font, keyboard, timezone, hostname and services.

Edit */etc/rc.d/net*, */etc/hosts* and */etc/resolv.conf* to configure your network (ip-

address/gateway/hostname/domain/dns).

Go to */usr/src/linux-2.6.xx.x*, configure and compile a new kernel. Useful kernel config files are placed into the */usr/src* directory. They are the config files used to build the boot kernels.

On some systems, you should apply the provided patches; these patches are found in the file */usr/src/*

GXT4000P/GXT6500P aren't supported by linux kernel then you have to apply our patch. After the installation process you'll find it as */usr/src/linux-2.6.xx.x.patch*. **CRUX PPC** provided patch enables support also for the IBM GXT4000P and GXT6500P display adaptor based on Raster Engine RC1000, found on some IBM System P (pSeries) machines. This driver doesn't use Geometry Engine GT1000.

```
$ cd /usr/src/linux-2.6.xx.x
$ make mrproper
$ cp ../linux-2.6.xx.x-Pseries.config .config
$ patch -p1 < ../cruxppc-logo.diff  (optional)
$ patch -p1 < ../linux-2.6.xx.x.patch
$ make menuconfig
$ make all
$ make modules_install
$ cp vmlinux /boot/vmlinux
$ cp System.map /boot/System.map
```

On pSeries computers, edit */etc/yaboot.conf* to boot the kernel you just compiled and type

```
$ dd if=/usr/lib/yaboot/yaboot.chrp of=/dev/sd??
```

where *sd??* is the device corresponding to the disk and PReP Boot partition. Remember that where */etc/yaboot.conf* resides must be the first Ext2fs formatted partition available on your disk.

It means you need to do a low-level copy of yaboot.chrp into your PReP partition and put your bootkernel and */etc/yaboot.conf* into your first ext2fs formatted partition.

e.g. */dev/sda3* as boot partition with ext2 filesystem:

```
root@power4:/home/acrux# fdisk -l /dev/sda
Disk /dev/sda: 73.4 GB, 73407820800 bytes
128 heads, 32 sectors/track, 35003 cylinders
Units = cylinders of 4096 * 512 = 2097152 bytes
Disk identifier: 0x416a4baa

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *           1           5       10224   41  PPC PReP Boot
/dev/sda2               6         518     1050624   82  Linux swap / Solaris
/dev/sda3             519         583      133120   83  Linux
/dev/sda4             584       35003    70492160    5  Extended
/dev/sda5             584        5704    10487792   83  Linux
/dev/sda6            5705       35003    60004336   83  Linux

root@power4:/home/acrux# df -Th
Filesystem    Type    Size  Used Avail Use% Mounted on
/dev/sda5 reiserfs     11G  2.5G  7.6G  25% /
/dev/sda6 reiserfs     58G  9.4G   48G  17% /home
/tmp         tmpfs    2.0G   14M  2.0G   1% /tmp
shm          tmpfs    2.0G     0  2.0G   0% /dev/shm
/dev/sda3     ext2    126M   37M   83M  31% /boot
```

```
root@power4:/home/acrux# ls -laR /boot
total 36013
drwxr-xr-x  4 root root     1024 Oct 10 22:28 .
drwxr-xr-x 20 root root      464 May 30 15:11 ..
-rw-r--r--  1 root root  1927086 Oct  4 14:43 System.map-2.6.30.8
-rw-r--r--  1 root root  1859378 Oct 10 22:58 System.map-2.6.31.3
-rw-r--r--  1 root root    40491 Oct  4 14:43 config-2.6.30.8
-rw-r--r--  1 root root    43607 Oct 10 22:58 config-2.6.31.3
drwxr-xr-x  2 root root     1024 Oct 10 22:28 etc
drwx------  2 root root    12288 May  6 19:40 lost+found
-rwxr-xr-x  1 root root 11090884 Oct  4 14:45 vmlinux-2.6.30.8
-rwxr-xr-x  1 root root 10041527 Oct 10 22:58 vmlinux-2.6.31.3
lrwxrwxrwx  1 root root       16 Oct 10 22:28 vmlinuz -> vmlinux-2.6.31.3
-rw-r--r--  1 root root      140 May  9 09:48 yaboot.msg

/boot/etc:
total 3
drwxr-xr-x 2 root root 1024 Oct 10 22:28 .
drwxr-xr-x 4 root root 1024 Oct 10 22:28 ..
-rw-r--r-- 1 root root  529 Oct 10 22:28 yaboot.conf

/boot/lost+found:
total 13
drwx------ 2 root root 12288 May  6 19:40 .
drwxr-xr-x 4 root root  1024 Oct 10 22:28 ..

root@power4:/home/acrux# cat /boot/etc/yaboot.conf
#
# yaboot-1.3.14 on pSeries 9114-275 (2-way POWER4+)
#

device=/pci@400000000110/pci@2,4/pci1069,b166@1/scsi@0/sd@3,0
partition=3

message=/yaboot.msg
delay=10

timeout=30
default=crux

image=/vmlinuz
      label=crux
      root=/dev/sda5
      append="video=gxt4500"
      read-only

image=/vmlinuz
      label=crux-ttyS0,9600
      root=/dev/sda5
      append="console=ttyS0,9600"
      read-only

image=/vmlinux-2.6.30.8
      label=old
      root=/dev/sda5
      read-only

enablecdboot
enableofboot

# EOF
```

Now remove the **CRUX PPC** CD-ROM from your drive and reboot from hard disk.

…going to tune our new CRUX PPC system

The first step is to tune */etc/pkgmk.conf* as we need and then rework */etc/ilenia.conf* . Many settings pertaining to the package build process can be adjusted by editing the pkgmk(8) configuration file */etc/pkgmk.conf* .

root@power4:/etc# cp pkgmk.conf pkgmk.conf.standard

A POWER4+ */etc/pkgmk.conf* and */etc/ilenia.conf* could be:

```
root@power4:/etc# cat /etc/pkgmk.conf
#
# /etc/pkgmk.conf: pkgmk(8) configuration
#

export CFLAGS="-O3 -mcpu=power4 -pipe -fsigned-char"
export CXXFLAGS="-O3 -mcpu=power4 -pipe -fsigned-char"
export FFLAGS="-O3 -mcpu=power4 -pipe -fsigned-char -funroll-loops"

export MAKEFLAGS="-j4"

# PKGMK_SOURCE_MIRRORS=()
# PKGMK_SOURCE_DIR="$PWD"
# PKGMK_PACKAGE_DIR="$PWD"
## PKGMK_WORK_DIR="/tmp/$name"
# PKGMK_WORK_DIR="$PWD/work"
# PKGMK_DOWNLOAD="no"
# PKGMK_IGNORE_FOOTPRINT="no"
# PKGMK_NO_STRIP="no"
# PKGMK_WGET_OPTS=""

# End of file
```

```
root@power4:/etc# cat ilenia.conf
#
# ilenia(8) configuration file
#

[ilenia]
POST_PKGADD = "pkgmk -c"
NOT_FOUND_POLICY = "Ask"
REPOSITORIES_HIERARCHY = "local/custom ppc/core ppc/opt ppc/xorg"
NEVER_INSTALL = "xorg-xf86-video-vesa elinks yaboot yasm nasm kqemu"
ASK_FOR_UPDATE = "Yes"
ENABLE_COLORS = "Yes"
# VERBOSE = "No"
# ENABLE_XTERM_TITLE = "Yes"
# DEFAULT_XTERM_TITLE = ""
# ENABLE_LOG = "Yes"
REJECTED_POLICY = "Check"

[favourite_repositories]
# Use:
# port = repository
#
# Using wildcard in port's names is allowed.

[locked_versions]
# Use:
# port = version

[aliases]
# Use:
# port = port1 port2 ....
#
# Note: port1 and port2 are aliases of port, but port isn't an alias for port1
# or port2, and also port1 isn't an alias of port2 and viceversa.
#

yaboot = yaboot-bin
```

```
icedtea6 = icedtea6-bin-ppc64
openoffice = openoffice-bin-ppc64

[pkgmk_confs]
# Use:

apache = /etc/pkgmk.conf.standard
*php*  = /etc/pkgmk.conf.standard
binutils = /etc/pkgmk.conf.standard

# End Of File
```

Now it's the time to update our ports tree with:

root@power4:~# ilenia -u

and then to update installed ports but, above all, install new ones that are strongly needed on a pSeries machine:

root@power4:~# ilenia -U powerpc-utils lsvpd

The last is to add new services to */etc/rc.conf*

```
root@power4:~# cat /etc/rc.conf
#
# /etc/rc.conf: system configuration
#

FONT=default
KEYMAP=us
TIMEZONE=UTC
HOSTNAME=power4
SYSLOG=sysklogd
SERVICES=(iprinitd rtas_errd inetd net crond dbus sshd apache ntpd)

#DISTCC_ALLOW="192.168.0.0/24"
# End of file
```

…advanced tuning and specific utilities

CRUX PPC is ready to use LVM2 features too. LVM2 refers to a new userspace toolset that provide logical volume management facilities on Linux. It is reasonably backwards-compatible with the original LVM toolset.

The Linux bonding driver provides a method for aggregating multiple network interfaces into a single logical "bonded" interface. This is primarily used for redundancy in ethernet paths or for load balancing. It's available in *ppc/opt* repository.

root@power4:~# ilenia -U ifenslave

To provides easy access to huge pages of memory is available libhugetlbfs in *ppc/opt* repository. It is a wrapper for the hugetlbfs file system. Applications can use huge pages to fulfill malloc() requests without being recompiled by using LD_PRELOAD.

root@power4:~# ilenia -U libhugetlbfs

If you need to work with Brain Fuck Scheduler (BFS) by Con Kolivas is also available schedtool in *ppc/opt* .

root@power4:~# ilenia -U schedtool

To increase **CRUX PPC** usability on pSeries we do provide ports for some IBM libraries

and utilities. We, also, do provide ports for Java Runtime Environment and Java Software Development Kit from IBM.

There are a number of small utilities for Linux on POWER that can come in useful for servicing are configuring your system. These first ones are available in *opt/ppc* repository.

root@power4:~# ilenia -U powerpc-utils

set_poweron_time: it can be used to specify a time in the future when the system or partition should be powered on.

bootlist: this command is used to modify the order of boot devices from the command line and the *ofpathname* utility is used by bootlist to convert between OF device names and logical device names.

usysident: is a tool for manipulating identification LEDs. These LEDs are used to help locate Field Replaceable Units (FRUs), to ensure that the correct part is being replaced. LEDs are specified by their location code or logical device name, and can be in one of two states: either “normal” (off) or “identify” (blinking amber LED). A related utility is *usysattn* that it’s used to turn off the system attention indicator, or to view the current state of that LED. The LED usually looks like an amber exclamation point located on the operator’s panel.

serv_config: is a very useful utility for modifying serviceability parameters in NVRAM.

uesensor: command can be used to view the values of various thermal, voltage, and fan speed sensors on the system. More recent systems will instead send an Environmental Or Power Warning (EPOW) event if any of the sensors are in danger of shifting out of the normal operating range.

nvram: is an utility to view and format the contents of the system's NVRAM.

snap: is a system configuration snapshot tool to gather system configuration data.

ppc64_cpu: is a cpu settings utility to view and set the cpu SMT, DSCR, and smt-snooze-delaysettings.

activate_firmware: is the concurrent firmware activation tool used for updating firmware.

hvcsadmin: HVCS driver administration utility

ibmvscsis.sh: IBMVSCSIS driver init script. When symlinked into start and stop run levels this script can provide vscsi server operation automation (not full tested on CRUX PPC).

rtas_dump: dump the contents of a rtas event, front end script for rtas_event_decode.

rtas_event_decode: workhorse for dumping the contents of a rtas event.

rtas_ibm_get_vpd: is used by the lsvpd utility to gather Vital Product Data.

update_flash: is a system firmware update utility

vscsisadmin: IBM Virtual SCSI Server (ibmvscsis) administration utility which simplifies configuration, starting, and stopping of an IBM virtual SCSI server (not full tested on CRUX PPC).

drmgr: Dynamic Resource Manager utility used to perform DLPAR operations on a client partition.

lsslot: an utility to list DLPAR and hotplug capable slots.

amsvis: this tool helps visualize metrics associated with Active Memory Sharing (AMS). The tool can be run locally but it can also be run remotely as either a data server or visualization client.

One nice advantage provided by POWER systems is the availability of structured and well-defined reference codes. Besides indicating errors or conditions that otherwise require attention, these codes are also used to indicate the progress of boots or dumps. If your system failed to boot for some reason, the last reference code on the operator's panel (op panel) would provide a good clue as to what the system was doing just before the failure.

The lsvpd package for Linux provides commands that can be used to retrieve an inventory of system hardware, along with the VPD associated with each device. The lsvpd package will install five commands: *lsvpd* ("list VPD"), *lscfg* ("list configuration"), *lsmcode* ("list microcode") *lsvio* ("list Virtual I/O") and *vpdupdate*.

root@power4:~# ilenia -U lsvpd

Unfortunately, the data in the lsvpd database can become stale as devices are added or changed (via hotplug or DLPAR, for example). Running *vpdupdate* will cause the data to be refreshed. The developers of lsvpd are currently working on having vpdudpate run automatically in response to hotplug events.

Hardware location codes provide a method for mapping connectors or logical functions to their physical locations in the system's enclosure. Starting with POWER5, both System p and System i started using the same format for location codes. The POWER4 (and POWER3 and JS20) location codes are much shorter (and differ between pSeries and iSeries), but you'll be able to deal with them once you get the hang of POWER5 (and later) location codes.

In case it matters, location codes are limited to 80 characters in length (to fit on the display on the operator's panel on the front of the system). They are composed only of uppercase characters, digits, periods, and dashes. They consist of labels separated by dashes. This is an example of a location code representing a PCI slot:

U787A.001.DNZ00Z5-P1-C3

The initial U indicates that the label represents a "unit" (like a CEC, central electronics complex, or an I/O drawer). The DNZ00Z5 at the end of the label is the serial number of the unit. The "P1" indicates the first planar on the system (i.e. the motherboard). The "C3" indicates the third PCI slot on that planar. Looking for labels on the system chassis will help you find the C3 slot, or the slot might also include an LED that can be turned on to identify the slot (with the usysident command, to be detailed in a near-future article).

Some of the other labels you may run into with location codes:

- An: the nth air handler (fan, blower, etc.)
- En: the nth electrical unit (power supply)
- Dn: the nth device (hard drive, SES device, etc.)
- Ln: the nth logical path (IDE address, SCSI target, fibre channel LUN, etc.)
- Tn: the nth port (on a multi-port ethernet adapter, for example)

Most PowerPC-based systems since POWER4 (including blades (JS20 and later)) have

dual firmware banks. The intent is to allow an administrator to update the system's firmware in one bank while preserving the previous firmware image in the other bank. Should there be a problem with the firmware update process, or an issue with the new image for any reason, the administrator can revert to the older, known-safe image.

During normal operation, the system is booted off of the "temporary" bank (sometimes called the temp side or t-side), and the contents of the temporary side are the same as that of the "permanent" bank (i.e. the perm side or p-side).

When a firmware update occurs, the new image is copied into the t-side. If the t-side image is different than the p-side image, the t-side will be copied to the p-side before the t-side is overwritten (i.e. the current production image will be backed up to the permanent bank). The system will attempt to reboot to the new image on the t-side. If the boot succeeds, and the new image works well, it can be "committed" to the p-side. If the system does not boot on the t-side (due to a corrupted image, for example), it will automatically boot onto the p-side. At that point, the t-side image can be "rejected" by overwriting it with the known-safe p-side image. The system should then be booted off of the t-side

The current firmware levels can be viewed by running *lsmcode -A*. The *update_flash* command provides the ability to manage system firmware from the Linux command-line on POWER systems.

References (URL's)

http://cruxppc.org/

http://zombieprocess.wordpress.com

[1] http://crux.nu/
[2] http://www.gnu.org/
[3] http://www.kernel.org/
[4] http://www.cruxppc.org/Main/Ilenia

Appendix #1: *about RISC Instruction Set Architecture*

A RISC (Reduced Instruction Set Computer) is an instruction set architecture (ISA) for the microprocessor that uses the reduced set of instructions both in size and complexity of the addressing modes thereby enabling easier implementation and the simultaneous memory operations in the CPU (by use of more registers and caches within CPU for higher operating frequencies) and facilitating use of the compilers efficiently. The RISC, or Reduced Instruction Set Computer, is an Instruction Set used by many current and former computer processors (or CPUs), including IBM POWER and CELL line of processors. The RISC instruction set focuses on simplified instructions, which allows the processor to run these instructions within a single clock cycle, thus traditionally allowing for greater performance.
Some of the typical examples where the RISC architecture based microprocessors are used are Alpha, MIPS, PA-RISC, SPARC, Power Architecture and others.

Appendix #2: *about Power Architecture®*

Power Architecture® technology is an instruction-set architecture for RISC microprocessors that spans applications from consumer electronics to supercomputers developed and manufactured by such companies as IBM, Freescale, AMCC, Tundra and P.A. Semi. The governing body is actually Power.org[1].
Power Architecture encompasses PowerPC®, Cell and POWERn™ processors. Scalability, flexibility and customization makes this microprocessing technology so different. Power is built on an open architecture, making it an open ecosystem that supports freedom of design in the marketplace.
Power Architecture is a family name too describing processor architecture, software, toolchain, community and end-user appliances and not a strict term describing specific products or technologies.

[1] http://www.power.org/

Appendix #3: *about this document*

This document is offered in good faith as comprising only safe programming and procedures. No responsibility is accepted by the author for any loss or damage caused in any way to any person or equipment, as a direct or indirect consequence of following these instructions.
The source of this 'in fieri' document is available here:

[1] http://cruxppc.org/~acrux/pseries.odt

CRUX PPC